IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 22 January 2023, accepted 8 February 2023, date of publication 13 February 2023, date of current version 17 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3244599

RESEARCH ARTICLE

Integrating Piecewise Linear Representation and Deep Learning for Trading Signals Forecasting

YINGJUN CHEN[1] AND ZHIGANG ZHU[2]
[1]Department of Computer Science and Technology, Tongji University, Shanghai 201804, China
[2]School of Health Science and Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China

Corresponding author: Yingjun Chen (superyjchen@126.com)

This work was supported by the University Capacity Building Project of the Shanghai Science and Technology Commission under Grant 21010502800.

ABSTRACT Trading signals forecasting is an interesting but challenging research topic in the field of financial investment, since the financial market is a nonlinearity and high volatility system influenced by too many factors, and a small improvement in forecasting performance can bring profits. To realize trading signals detection, this paper presents a novel method which integrates piecewise linear representation (PLR) with a deep learning framework to predict the financial trading points. Firstly, we utilize PLR to generate a number of turning points (valleys or peaks) from trading data and formulate the trading points prediction as a three-class classification problem. Then, the framework combined a convolutional neural network (CNN) for spatial features extraction and a long short-term memory (LSTM) network for temporal domain features extraction (CNN-LSTM) is used to learn the prediction model between the trading points and the financial time series data. Finally, we conduct a series of experiments among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM on companies of US, Turkey and daily Exchange-Traded Fund (ETFs) to test the performance of our established method. The experiment results show that our proposed method has better model performance and profitability with different investment strategies.

INDEX TERMS Piecewise linear representation (PLR), convolutional neural network (CNN), long short-term memory (LSTM), trading signals detection.

I. INTRODUCTION

Trading signals forecasting is one of the most attractive but challenging research topics in the field of financial investment, since the financial market is an unstable, non-linear and complex system, and a small performance improvement in trading point forecasting can be profitable. A lot of investors and researchers have tried to find the most suitable trading points, but inevitably made some wrong decisions. The reason is that the trading points are affected by many interrelated factors, such as the change of national policies, domestic and foreign economic environments, political situations, international situations, psychological variables of investors and so on [1], [2]. However, maximizing the benefits and reducing investment risks through reasonable and accurate forecasting of trading points has motivated researchers to develop more effective forecasting techniques.

For the techniques used for decision-making in financial markets, professional traders employ fundamental analysis, technical analysis and artificial intelligence methods. Fundamental analysis uses macroeconomic, industrial and business indicators to make trading decisions [3], [4]. Technical analysis is based on the assumption that past behavior has an impact on future price evolution, and trading decisions are made based on historical market prices and technical indicators such as moving averages, bollinger bands, stochastic oscillators, and so on [5], [6], [7], [8]. Generally speaking, financial time series data are inherently chaotic, noisy, and nonlinear, and do not necessarily follow a fixed pattern. Therefore, fundamental analysis and technical analysis are not good at predicting trading signals. In contrast, artificial intelligence methods can handle random, chaotic,

The associate editor coordinating the review of this manuscript and approving it for publication was Mauro Tucci.

and nonlinear data in the financial market and are widely used to predict trading signals and make buying and selling decisions.

Over the decades, many artificial intelligence algorithms have been developed and applied to financial market forecasting, for instance, artificial neural network [9], [10], [11], support vector machines [12], [13], [14], [15], [16], [17], rough set theory [18], [19], [20], bayesian analysis [21], [22], [23], [24] and evolutionary learning algorithms [25], [26], [27], [28]. However, most of the past researches mainly focus on the accurate price forecast only. In the field of stock forecasting, the most important goal is to obtain high and stable profits. Therefore, how to predict the trading points and provide investors with effective trading methods to obtain high and stable profits in the financial practice is particularly important. To achieve this goal, this research develops a novel intelligent trading signals detection method to detect the trading points in the financial market more effectively.

In recent years, the prediction and classification performance of deep learning has been getting better and better, and it has begun to be widely used in various fields, slowly surpassing the traditional computational intelligence methods [29], [30], [31], [32], [33], [34]. However, deep learning models are mainly used in fields such as image and video recognition [35], [36], [37], speech recognition [38], [39], [40], natural language processing [41], [42], and expert systems [43], [44]. In recent years, deep learning methods have begun to appear in financial research [45], [46], [47], [48]. However, the application of deep learning in financial forecasting models is very limited. Sezer and Ozbayoglu proposed a algorithmic trading model CNN-TA, which used a 2-D convolutional neural network through image processing properties [45]. Hoseinzade et al. presented a framework called U-CNNpred, that used a CNN-based structure, which was trained in a specially designed layer-wise training procedure over a pool of historical data from many financial markets [49]. Kelotra and Pandey proposed a system named Rider-monarch butterfly optimization (MBO)-based deep-convolutional long short-term memory (ConvLSTM) model for predicting the state of the stock market [50].

Convolutional neural networks (CNNs) [51] are good at dealing with locally related data in adjacent locations and Long Short-Term Memory (LSTM) is a time-sequential model [52], which can extract the temporal domain features from any sequential data. Inspired by the success of deep learning in various fields, we proposes a new hybrid deep learning trading signal forecasting framework PLR-CNN-LSTM.

The contributions of this research are as follows:

(1) We generate numerous trading points (valley or peak) from the trading data by utilizing PLR method and formulate the stock trading signals prediction as a three-class (Buy/Sell/Hold) classification problem. Then, we convert one-dimensional financial time series data into a two-dimensional image representation.

(2) We develop a deep learning model, which uses CNN to extract spatial features and uses LSTM to extract the temporal domain features, to analyze the nonlinear relationships between trading signals and various inputs, and to capture the knowledge of trading signals that are hidden in historical data. Then, the learned model is used to predict the future trading signals.

(3) We investigate two different investment strategies, which are the strategy for investors possessing a lot of funds and the strategy for investors with limited investment capital in this study.

As a result of these contributions, we observed that the proposed prediction model can be applied to forecast the stock trading signals in the real-world application.

The rest of this paper is organized as follows. In Section II, we describe the data description, which includes raw data description, input variable selection, data preprocessing, data labeling and images generation. In Section III, we briefly illustrate the technologies of CNN, LSTM, CNN-LSTM, performance measure and the framework of PLR-CNN-LSTM. Section IV presents some experiment results to validate the performance of our proposed method. Finally, concluding this work and some brief comments are presented in Section V.

II. DATA DESCRIPTION

A. RAW DATA

In this paper, the datasets utilized for this study involve companies of US (Developed Markets), Turkey (Emerging Market) and the daily Exchange-Traded Fund (ETFs) for experimental purpose.

Let S be the stock or ETfs dataset, described as follows.

$$S = \begin{bmatrix} s_{1,open} & s_{1,low} & s_{1,high} & s_{1,close} & v_1 \\ s_{2,open} & s_{2,low} & s_{2,high} & s_{2,close} & v_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_{N,open} & s_{N,low} & s_{N,high} & s_{N,close} & v_N \end{bmatrix} \quad (1)$$

where $s_{i,open}$, $s_{i,low}$, $s_{i,high}$, $s_{i,close}$ and v_i $(i = 1, 2, \ldots, N)$ represent the opening price, the lowest price, the highest price, the closing price and the trading volume of the index for i^{th} day respectively.

B. INPUT VARIABLE SELECTION AND DATA PREPROCESSING

In the financial market prediction problem, historical opening price, closing price, highest price, lowest price and volume are usually regarded as the input variables. Recently, some experts and researchers have shown that technical indicators are important features for modeling in financial market [53], [54], [55]. They are effective tools to characterize the real market situation in financial time series prediction [2] and can be more informative than pure prices [56]. Consequently, based on the review of domain experts and literatures, many important technical indicators will be taken into consideration along with the daily values. These technical indicators are of any class of metrics whose values are calculated by applying a

formula to the basic daily values. A brief explanation of each indicator is provided here.

(i) Kaufman Adaptive Moving Average (KAMA)

KAMA is an intelligent moving average. This powerful trend-following indicator is based on an exponential moving average (EMA) that responds to both trends and volatility. It follows the price when the noise is low, and smoothes out the noise when the price fluctuates. Like all moving averages, KAMA can be used to visualize trends.

(ii) Exponential Moving Average (EMA)

$$EMA_n = \begin{cases} s_{1,close} & if\ n = 1 \\ \frac{2}{n+1} * s_{n,close} + \frac{n-1}{n+1} * EMA_{n-1} & if\ n > 1 \end{cases} \quad (2)$$

EMA is also an weighted average of the fixed subset, which focuses more on most recent prices rather than on long series.

(iii) Moving Average Convergence / Divergence (MACD)

$$\begin{aligned} DIF_n &= EMA_{n_{fast}} - EMA_{n_{slow}} \\ DEA_n &= \alpha * DEA_{n-1} + (1-\alpha) * DIF_n \\ MACD_n &= 2 * (DIF_n - DEA_n) \end{aligned} \quad (3)$$

MACD is a trend-following momentum indicator, which is a collection of three time series calculated from historical prices and shows the relationship between a fast and slow exponential moving average.

(iv) Simple Moving Average (SMA)

$$SMA_i(n) = \frac{1}{n} \sum_{j=\max\{1,i-n\}}^{i} s_{j,close} \quad (4)$$

SMA is a calculation that takes the arithmetic mean of a given set of prices over the specific number of days in the past.

(v) Relative Strength Index (RSI)

$$RSI_n = 100 - \frac{100}{1 + EMA_{n_{up}}/EMA_{n_{down}}} \quad (5)$$

where $EMA_{n_{up}}$ is upward change, $EMA_{n_{down}}$ is downward change. RSI is an extremely popular momentum indicator that measures the speed and the change of price movements. It is usually interpreted as an overbought/oversold indicator.

(vi) Weighted Moving Average (WMA)

WMA puts more emphasis on recent data and considers older data less important. To achieve this, the mean is multiplied by the value of each bar with a certain weighting factor.

(vii) Directional Movement Index (DMI)

DMI is a technical indicator that measures both the strength and direction of a price movement and is intended to reduce false signals.

(viii) Triple Exponential Moving Average (TEMA)

TEMA was designed to smooth price fluctuations, thereby making it easier to identify trends without the lag associated with traditional moving averages. It does this by taking multiple exponential moving averages of the original EMA and subtracting out some of the lag. TEMA uses multiple EMA calculations and subtracts out the lag to create a trend following indicator that reacts quickly to price changes.

(ix) Average True Range (ATR)

$$ATR_n = EMA_n(max(s_{i,high} - s_{i,low}, |s_{i,high} - s_{i,close} - 1|, |s_{i,low} - s_{i-1,close}|)) \quad (6)$$

where $s_{i,high}$, $s_{i,low}$ and $s_{i,close}$ are the highest, lowest and closing prices on day i respectively, $|\ldots|$ denotes the absolute value, and n is the input window length. ATR provides information about the degree of price volatility.

(x) Commodity Channel Index (CCI)

$$CCI_n = \frac{sum^t - SMA_n(sum^t)}{0.015 \sum_{i=1}^{n} |sum^{t-i+1} - SMA_n(sum^t)|/n} \quad (7)$$

where sum^t is a sum of the highest, lowest and closing prices on t^{th} day. CCI is an oscillator used to determine whether a stock is overbought or oversold.

(xi) Price rate-of-change (ROC)

$$ROC_n = \frac{s_{i,close} - s_{i-n,close}}{s_{i-n,close}} \quad (8)$$

ROC shows the relative difference between the closing price on the i^{th} day of forecast and the closing price before n days.

(xii) The William's %R oscillator

$$Williams\_R_n = \frac{100 * (s_{n,high} - s_{n,close})}{(s_{n,high} - s_{n,low})} \quad (9)$$

It shows the relationship between the current closing price and the highest and lowest prices over the latest n days equal to the input window length.

(xiii) Momentum (MOM)

$$MOM_n = \frac{s_{i,close}}{s_{i-n,close}} \quad (10)$$

The Momentum is a measurement of the acceleration and deceleration of prices. It indicates that prices are increasing at an increasing rate or decreasing at a decreasing rate. The Momentum function can be applied to price or any other data series.

(xiv) Average Directional Movement Index (ADX)

$$ADX_n = \frac{ADX_{n-1} * (N-1) + DX}{n} \quad (11)$$

The ADX is a Welles Wilder style moving average of the Directional Movement Index (DX). To interpret the ADX, consider a high number to be a strong trend, and a low number to be a weak trend.

(xv)Chande Momentum Oscillator (CMO) The Chande momentum oscillator is a technical momentum indicator. The formula calculates the difference between the sum of recent gains and the sum of recent losses and then divides the result by the sum of all price movements over the same period.

After adding technical indicators, now the new dataset S is reformulated as follows (12), shown at the bottom of the next page.

To avoid variables in greater numeric ranges dominating those in smaller numeric ranges and numerical difficulties during calculation, the selected input variables are scaled

between 0 and +1 by using the standard formula described as follows.

$$\widehat{s}_{i,j} = \frac{s_{i,j} - \min s_j}{\max s_j - \min s_j} \tag{13}$$

where $s_{i,j}$ is the current day value, $\widehat{s}_{i,j} \in [0, 1]$ is the scaled value, $\min s_j = \min\{s_{i,j}, i = 1, 2, \ldots, N\}$ is the minimum value of j^{th} feature of S, and $\max s_j = \max\{s_j, i = 1, 2, \ldots, N\}$ is the maximum value of j^{th} feature of S.

C. DATA LABELING

A trading signal is a local peak or valley value during a price movement. A valley can be defined as a buy point, and a peak can be defined as a sell point. How to generate trading signals is the first problem that needs to be solved. Based on the review of domain experts and prior researches [57], [58], this study applies piecewise linear representation (PLR) to decompose historical time-series data for stocks into distinct segments, so that trading signals are junctions between adjacent segments.

PLR is developed for pattern matching, which is one of the most commonly used piecewise linear approximation representations method. In financial time series field, many researchers and investigators use PLR to generate turning points [59], which usually represent trading signals. Let $X = \{x_1, x_2, \ldots, x_N\}$ denote the financial time series data, which can be decomposed into M segments by the piecewise approximation straight lines. The generated segments are described by as follows.

$$S_{PLR} = \{L_1(x_1, x_2, \ldots, x_{t_1}), L_2(x_{t_1+1}, x_{t_1+2}, \ldots, x_{t_2}), \ldots, L_M(x_{t_{M-1}+1}, x_{t_{M-1}+2}, \ldots, x_{t_M})\} \tag{14}$$

where $t_i (i = 1, 2, \ldots, M)$ denotes the end of the i^{th} segment and also the change of the movement trend. Thus, the end points from PLR can be the turning points.

Most of the time series segmentation methods can be divided into three types, sliding windows method, top-down method and bottom-up method [58]. In this paper, we use top-down algorithm to segment the financial series data and generate the trading signals. The reason it that top-down algorithm can produce a better representation in any segment where the maximum error does not exceed a given threshold. However, the threshold value of a piecewise representation is the key factor affecting the representation of the segmentation. Figure 1, Figure 2 and Figure 3 shows the trading signal segmentation graph of DJIA index under different thresholds, where the time span for DJIA index is from Feb. 1st 2001 to Jun. 25th 2003. Each segment

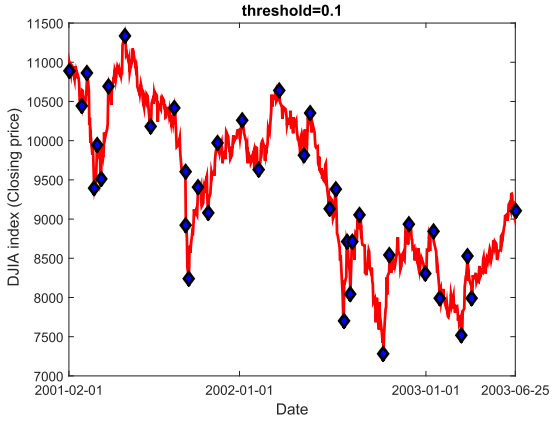

FIGURE 1. The turning points obtained by PLR with the threshold value of 0.1.

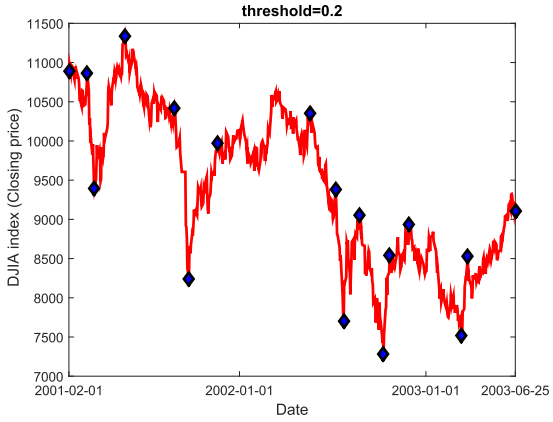

FIGURE 2. The turning points obtained by PLR with the threshold value of 0.2.

in the graph represents a current valley or peak which is converted into trading signals. As shown in the figure, the higher threshold value produces fewer segments, while the smaller value produces more segments. There are roughly 38 turning points for the threshold value of 0.1, while there are only 12 turning points for a threshold value of 0.3. However, there is no guarantee regarding to which value will be better. Therefore, it is important to choose a suitable threshold for each financial time series data. In this paper, we utilize algorithm SD [58] to automatically select this threshold, in which different values for different series data or different price fluctuations are specified in accordance with the parameter *pct*.

The turning points generated by utilizing PLR are divided into three classes in order to generate the class labels. The

$$S = \begin{bmatrix} s_{1,open} & s_{1,low} & s_{1,high} & s_{1,close} & TI_{1,1} & \cdots & TI_{1,m} \\ s_{2,open} & s_{2,low} & s_{2,high} & s_{2,close} & TI_{2,1} & \cdots & TI_{2,m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ s_{N,open} & s_{N,low} & s_{N,high} & s_{N,close} & TI_{N,1} & \cdots & TI_{N,m} \end{bmatrix} \tag{12}$$

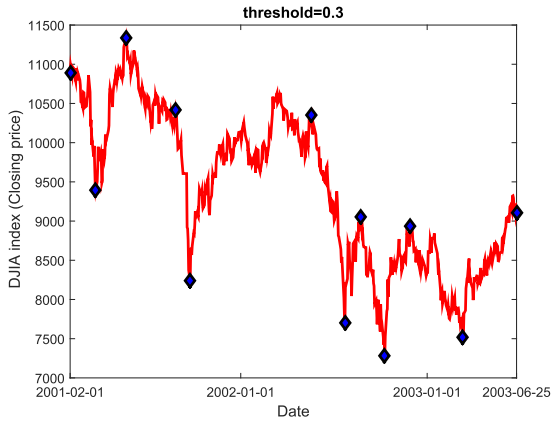

FIGURE 3. The turning points obtained by PLR with the threshold value of 0.3.

turning points with valley is labeled as buying point (BP), while peak is labeled as selling point (SP). The other points are labeled holding point (HP). HP, BP, SP are numbered as 0, 1,and 2, respectively. After generating the class labels $y_i \in \{0, 1, 2\}$, the dataset S is reformulated as follows (15), shown at the bottom of the page.

D. IMAGES GENERATION

In order to be able to utilize the power of deep learning for trading signals forecasting, we generate a two-dimensions image-like data in our proposed forecasting framework. Therefore, we select 15 medium and short term technical indicators related to volatility, and use different parameter settings for each technical indicator to generate a two-dimensional image representation. Each row of the x-axis in the two-dimensional image consists of a sequence of 15 technical indicators at a specific time point, and each column of the y-axis is generated by 20-day parameter settings. To satisfy the requirement of locality and provide a consistent and meaningful image representation, similar technical indicators are clustered together along the x-axis. Therefore, a 20×15 pixel image is generated for each day's data by calculating the technical indicators of 3-23 days span. The reason why we use the 3-23 days indicator range in our study is that the volatility trading from 1 week to 1 month is concentrated in the 3-23 days' time range. According to the research of this paper, it is novel to represent financial time series data as images according to local features, and use them as the input of CNN to predict financial time series data. Figure 4 shows the 20×15 pixel image representation created during the image generation phase.

FIGURE 4. Generated images in images generation phase.

III. METHODOLOGY

After the turning points are generated, how to model the relationship between input variables and trading signals to realize the detection of trading signals is the most important problem to be solved. In this paper, we propose a novel prediction algorithm CNN-LSTM, which integrates CNN and LSTM to predict financial trading signals.

A. THEORY OF CNN

CNN is currently the most commonly used deep learning model, which is a feed-forward neural network and the input is a matrix or vector [60], [61]. Different from MLP and other fully connected neural networks, the locality of data within the input is very important in CNN. Therefore, when we are solving practical problems with CNN, the neighboring data points within the matrix should be carefully chosen. CNN was initially widely used in image and video recognition, speech recognition, natural language processing and expert system due to the correlation on the input data in these fields, which directly meets the characteristics of CNN [62], [63].

However, not all problems are as easy to be optimized as picture classification problems in deep learning. In order to better solve financial time series prediction using deep learning, we do not stack each CNN layer directly, but make these layers fit residual mapping by introducing deep residual learning framework as which can learn the constant transformation more easily by skip-connections [64], [65], [66].

As shown in Figure 5, suppose $H(x)$ is a fit that will be mapped through several stacked network layers, where x is the input of the first layer of these layers, and $H(x)$ and x have the same dimension. Assuming that multiple nonlinear layers can asymptotically approximate the complex function $F(x)$, it is equivalent to assuming that $F(x)$ can asymptotically approximate the residual function $H(x) - x$. So we expect these stacked layers will approximate $F(x) = H(x) - x$, but not $H(x)$. Therefore, the original function becomes $F(x) + x$. Although it is possible to asymptotically approach

$$S = \left[\begin{array}{cccc:cccc} s_{1,open} & s_{1,low} & s_{1,high} & s_{1,close} & TI_{1,1} & \cdots & TI_{1,m} & y_1 \\ s_{2,open} & s_{2,low} & s_{2,high} & s_{2,close} & TI_{2,1} & \cdots & TI_{2,m} & y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ s_{N,open} & s_{N,low} & s_{N,high} & s_{N,close} & TI_{N,1} & \cdots & TI_{N,m} & y_N \end{array}\right] \tag{15}$$

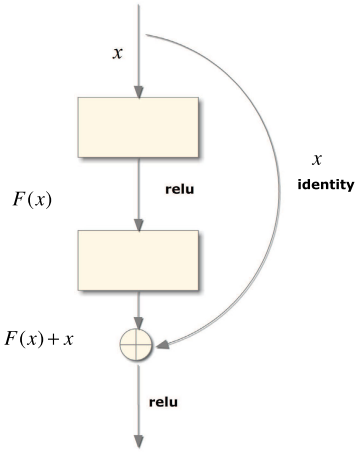

FIGURE 5. RESNET structure.

the target function in either form, the difficulty of the learning may differ. The identity shortcut connection neither adds additional parameters nor increases computational complexity and the entire network can still be trained end-to-end by SGD with backpropagation.

B. THEORY OF LSTM

In this section, a brief introduction of LSTM is provided, the details of which can be found in [14]. LSTM is a neural network for processing sequence data. Compared with the general neural network, it can process the data of sequence change and extract the temporal domain features from any sequential data. It can also solve the problem of gradient disappearance and gradient explosion during long sequence training. Figure 6 shows the cell structure of the LSTM model. It consists of a memory block used to contain a memory cell and three gates, which are the input, output, and forget gates to control that cell.

1) FORGET GATE

The forget gate is a key component of an LSTM cell that controls what information to keep and what to forget. It can also avoid the vanishing and exploding gradient problems that arise when gradients backpropagate over time. The formulation is as follows:

$$f_t = \sigma(w_f \cdot [h_{t-1}, x_t] + b_f) \tag{16}$$

where σ is the activation function, w_f is the weight of the forget gate, b_f is the bias of the forget gate, x_t is the input value of the current time, h_{t-1} is the output value of the last moment.

2) INPUT GATE

The input gate is used to control how much the current input data of the network flows into the memory unit, that means how much input information can be saved. The input gate consists of two parts. The first part generates the control signal between 0 and 1 by the input gate composed of sigmoid, which is used to control the degree of input data.

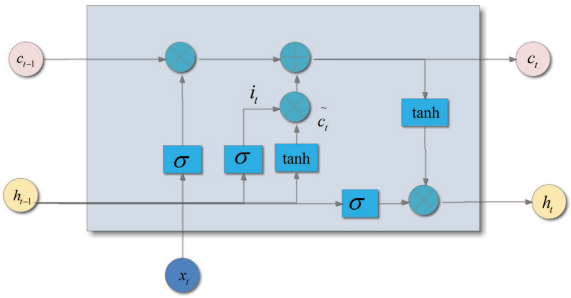

FIGURE 6. LSTM structure.

The second part generates the current moment through a tanh layer. The formulations are as follows:

$$i_t = \sigma(w_i \cdot [h_{t-1}, x_t] + b_i) \tag{17}$$

$$\tilde{C}_t = \tanh(w_C \cdot [h_{t-1}, x_t] + b_C) \tag{18}$$

where w_i is the weight of the input gate, b_i is the bias of the input gate, w_C is the weight of the candidate input gate, and b_C is the bias of the candidate input gate.

3) UPDATE CELL STATE

Update the old cell state to the current cell state. With the control signal generated by the forget gate, the candidate cell state generated by the tanh layer, and the control signal generated by the input gate, the cell can be updated as follows.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{19}$$

where C_{t-1} is the cell state of the previous moment, i_t is the output of input gate for the current moment and $\tilde{C}_t$ is the candidate cell status at the current moment.

4) OUTPUT GATE

The output value is based on the cell state, but there will be a filtering process. This also includes two parts. The first part is used to generate the control signal between 0 and 1. The second part is that the final output information is multiplied by the control signal to get the final output value.

$$o_t = \sigma(w_o \cdot [h_{t-1}, x_t] + b_o) \tag{20}$$

$$h_t = o_t * \tanh(C_t) \tag{21}$$

where w_o is the weight of the output gate, b_o is the bias of the output gate and C_t is the cell state at the current moment.

C. THEORY OF CNN-LSTM

This section briefly describes the architecture of our proposed CNN-LSTM model. Figure 7 depicts the overall CNN-LSTM model architecture for predicting financial time series data. The CNN network structure in CNN-LSTM refers to the residual learning theory, which is used to extract spatial features. The input layer produces 20×15 images that is generated by 15 separate technical indicators with different time intervals. In CNN, we use 3×3 convolution kernel. The reason is that we have relatively small images (20×15), which can have significant changes in intensity.

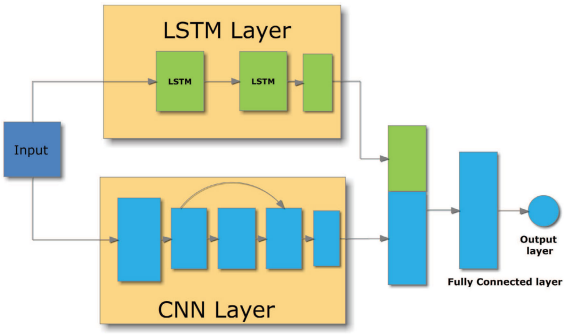

FIGURE 7. CNN-LSTM structure.

As the most commonly used and effective convolution kernel on small images, 3×3 convolution kernel provides the most intimate fields in the process of processing the convolution layer, so drastic changes within the image can be captured. The CNN also adds a dropout layer to prevent over-fitting. In the CNN model, more layers are not added for keeping the simplicity of the network. Without a large training set, a large and complex network is likely to overfit and reduce the generalization ability of the model. For future, deeper models with more processing layers can be designed as more training data becomes available.

In the sequence learning block, we use two LSTM layers with 64 neurons each. The return sequence is set to true for the fisrt LSTM layer so that the network will output the full sequence of hidden states. Whereas the return sequence is set to false in the final LSTM layer so that the network will output the hidden state at the final time step. We use the dropout layer after the final LSTM layer. The output of LSTM layer is concated with the output of CNN structure, then the concated output is connected a fully connected layer.

D. PERFORMANCE MEASURE

The overall performance of our proposed model is evaluated using two different evaluation criteria: computational model performance and financial evaluation. Computational performance evaluation presents the model performance, i.e. how well the classifier distinguishes between Buy, Hold and Sell classes. Financial evaluation shows the performance of the whole proposed model by implementing the real world financial scenario. Stocks or ETFs are bought, sold or held according to the predicted label with the actual prices.

In trading signal prediction, we are concerned about the recalling situation of Buy and Sell signals. It directly affects whether the trading system buys and sells at the right time. In machine learning, there are two categories (positive, negative) of binary problem samples. Then, there are four combinations of model predicted results and real labels: TP, FP, FN, TN.

(1) TP – the number of positive classes predicted as positive classes;

(2) FN – the number of positive classes predicted to be negative classes;

(3) FP – the number of negative classes predicted to be positive classes;

(4) TN – the number of negative classes predicted to be negative classes;

The commonly used evaluation for binary classification problem are precision, recall, F1 score and accuracy. The definitions are as follows:

$$precision = \frac{TP}{TP + FP} \tag{22}$$

$$recall = \frac{TP}{TP + FN} \tag{23}$$

$$F1\ score = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{24}$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{25}$$

The most important target for trading signal prediction is to make a profit by the algorithm. In order to evaluate the performance of our proposed PLR-CNN-LSTM method objectively, we adopt two investment strategies to investigate the profitability. Let $b_{stock}(i)$ denote the balance number of stock for i^{th} day, $b_{money}(i)$ denote the balance money for i^{th} day, $v_{money}(i)$ denote total investment money until i^{th} day.

Strategy A: We use this strategy to investigate the profitability for the investors possessing a lot of funds.

(a) Buying strategy: if y_i is 1 on i^{th} day, the investors buy one unit. The balance number, total investment money, and the balance money are calculated according to (26), (27), (28) respectively.

$$b_{stock}(i) = \begin{cases} 1 & if\ i = 1 \\ b_{stock}(i-1) + 1 & if\ i > 1 \end{cases} \tag{26}$$

$$v_{money}(i) = \begin{cases} s_{1,close} & if\ i = 1 \\ v_{money}(i-1) & if\ 0 \le b_{money}(i-1) \\ +(s_{i,close} - b_{money}(i-1)) & \le s_{i,close}, i > 1 \\ v_{money}(i-1) + s_{i,close} & if\ b_{money}(i-1) > s_{i,close}, \\ & i > 1 \end{cases} \tag{27}$$

$$b_{money}(i) = \begin{cases} 0 & if\ i = 1 \\ b_{money}(i-1) - s_{i,close} & if\ b_{money}(i-1) > s_{i,close}, i > 1 \\ 0 & if\ 0 \le b_{money}(i-1) \le s_{i,close}, \\ & i > 1 \end{cases} \tag{28}$$

(b) Selling strategy: if $y_i = 2$ and $b_{stock}(i) > 0$ on i^{th} day, the investors always sell all their stocks. When all stocks are sold, the balance money and the balance number are calculated according to (29), (30) respectively.

$$b_{money}(i) = b_{money}(i-1) + b_{stock}(i) \times s_{i,close} \tag{29}$$

$$b_{stock}(i) = 0 \tag{30}$$

where $s_{i,close}$ is the actual closing price on i^{th} day.

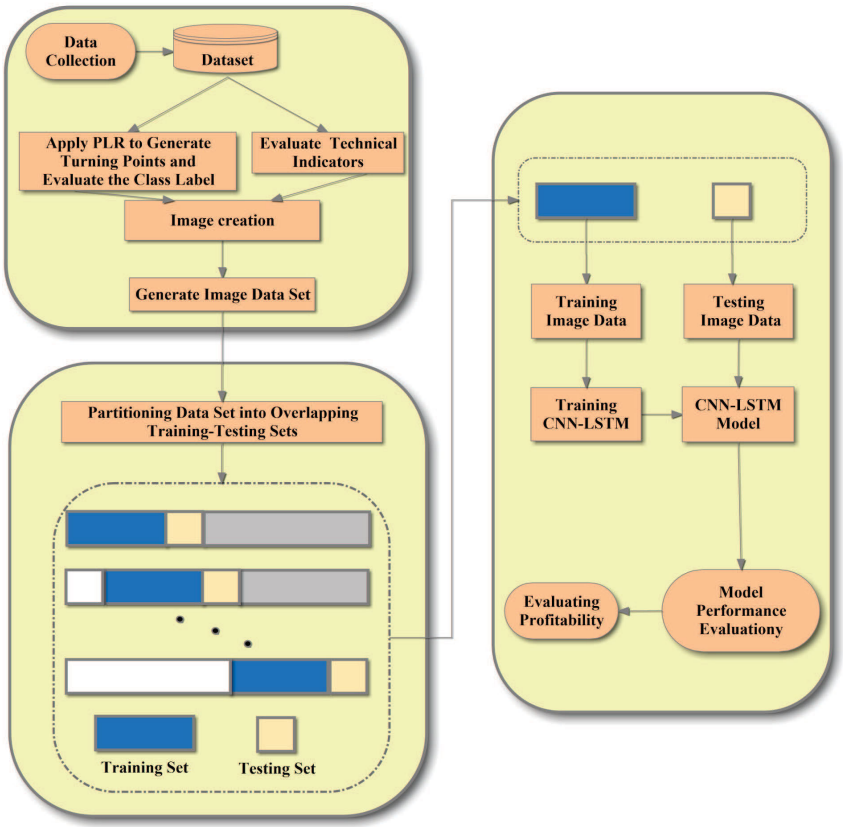

FIGURE 8. Schematic layout of PLR-CNN-LSTM for financial markets analysis.

Strategy B: We use this strategy to investigate the profitability for retail investors. The total investment capital is limited to 100000.

(a) Buying strategy: if $y_i = 1$ and $b_{money}(i) > 0$, the investors buy stock with the balance money. After buying, the balance number and money are calculated according to (31), (32) respectively.

$$b_{stock}(i) = \begin{cases} \dfrac{100000}{s_{1,close}} & if\ i = 1 \\ b_{stock}(i-1) + \dfrac{b_{money}(i)}{s_{i,close}} & if\ i > 1 \end{cases} \tag{31}$$

$$b_{money}(i) = 0 \tag{32}$$

(b) Selling strategy: if $y_i = 2$ and $b_{stock}(i) > 0$ on i^{th} day, the investors always sell all their stocks. The balance money and the balance number for i^{th} day are updated according to (29)-(30).

At the end of an investment period, all stocks must be sold at the closing price of the last day. The profit for strategy is computed by

$$profit_{money} = \frac{b_{money} - v_{money}}{v_{money}} \tag{33}$$

E. SCHEMATIC LAYOUT OF PROPOSED PLR-CNN-LSTM

This section outlines the trading signal prediction framework based on PLR-CNN-LSTM. As illustrated in Figure 8, this framework consists three basic phases: (1) the data collection, turning points generating by PLR, and image generation phase (2) partitioning the whole dataset into overlapping training and testing (3) CNN-LSTM model training and testing.

In phase 1, the opening price, the lowest price, the highest price and the closing price are first collected and stored in the database. Then, PLR is used to segment the financial series data into the peak and trough to generate the label y_i. Lastly, the images data is generated according to the image generation method, the images and label y_i are added to the initial dataset.

In phase 2, we divide the whole dataset into some overlapping training-testing sets. in order to reduce the time-varying characteristics of financial transaction data, which are marked as S_{train} and S_{test} respectively. Let L_{train} denote the size of training set S_{train}, L_{test} denote the size of testing set S_{test}, L_S denote the size of the whole data set S. The whole data set will be divided into M overlapping training-testing sets in accordance with $M = \lceil \frac{L_S - L_{train}}{L_{test}} \rceil$, where $\lceil x \rceil$ denotes the minimal positive integer that is not less than x.

In phase 3, Firstly, we construct a three-class classification problem for each overlapping training-testing set $S^i = S^i_{train} \cup S^i_{test}, i = 1, 2, \ldots, M$. We train CNN-LSTM on the training data and export the trained model for testing. In the stage of testing, we firstly classify the testing data using the trained model and compute model performance. Then, we adopt two

TABLE 1. Description of stocks data samples and data range.

Country	companies	Data range
US	Intel, Netflix, Disney	1/Jan/2013 to 30/Dec/2022
Turkey	Arcelik, Dogus, Koc Holding A.S	1/Jan/2013 to 30/Dec/2022

TABLE 2. Description of ETFs data samples and data range.

Name	Description	Data range
QQQ	PowerShares QQQ ETF	1/Jan/2013 to 30/Dec/2022
SPY	SPDR S&P 500 ETF	1/Jan/2013 to 30/Dec/2022
XLP	Consumer Staples Select Sector SPDR ETF	1/Jan/2013 to 30/Dec/2022
XLU	Utilities Select Sector SPDR ETF	1/Jan/2013 to 30/Dec/2022

TABLE 3. Description of technical indicators.

No.	Technical indicator	No.	Technical indicator	No.	Technical indicator
1	RSI	6	TEMA	11	ROC
2	Williams	7	KAMA	12	DMI
3	WMA	8	CCI	13	ATR
4	EMA	9	CMO	14	ADX
5	SMA	10	MACD	15	MOM

TABLE 4. The CNN-LSTM model parameters.

Layer (type)	Output shape	Number of parameters
input_1 (InputLayer)	(None, 20, 15, 1)	0
reshape_2 (Reshape)	(None, 20, 15, 1)	0
conv2d_1 (Conv2D)	(None, 18, 13, 24)	240
conv2d_2 (Conv2D)	(None, 16, 11, 16)	3472
pooling2d_1 (MaxPooling2D)	(None, 16, 11, 16)	0
conv2d_3 (Conv2D)	(None, 16, 11, 16)	1040
conv2d_4 (Conv2D)	(None, 16, 11, 16)	1040
reshape_1 (Reshape)	(None, 20, 15)	0
add_1 (Add)	(None, 16, 11, 16)	0
lstm_1 (LSTM)	(None, 20, 64)	20480
dropout_3 (Dropout)	(None, 16, 11, 16)	0
dropout_1 (Dropout)	(None, 20, 64)	0
flatten_1 (Flatten)	(None, 2816)	0
lstm_2 (LSTM)	(None, 64)	33024
dense_1 (Dense)	(None, 400)	1126800
dropout_2 (Dropout)	(None, 64)	0
concatenate_1 (Concatenate)	(None, 464)	0
dense_2 (Dense)	(None, 256)	119040
dropout_4 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 3)	771

investment strategies to evaluate the financial performance of the proposed model.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

A. DATA COLLECTION AND EXPERIMENTAL SETUP

To investigate the performance of PLR-CNN-LSTM, this study considers the three performing companies of US (Developed Markets), Turkey (Emerging Market) and the four daily Exchange-Traded Fund (ETFs). The collected stocks datasets are depicted in Table 1, where the time span for these stocks is from Jan. 1st 2013 to Dec. 30th 2022, there are about ten years of data. Selected ETFs and their descriptions are illustrated in Table 2. These six stocks and four ETFs include emerging markets, developed markets and the highest trading volumes daily Fund, providing us with a natural environment to test the robustness of model performance under different market conditions.

In the experiments, the Keras framework is used. The network parameters are optimized by using stochastic gradient descent. The learning rate is 0.1, the decay over each update is 0.001 and Nesterov momentum is 0.9. The size of each training set is 1000, the size of each testing set is 200. That means our proposed model is trained with 1000 training data and tested with 200 out-of-sample data. Then, the network is retrained with the next 1000 training data which is generated by moving ahead 200 data and tested with next 200 out-of-sample data. The parameter *pct* in algorithm SD is 0.5. There are 15 technical indicators in total, and the details are shown in Table 3. These variables are correlated with variations in stock prices to some degree. The proposed PLR-CNN-LSTM model architecture and parameters is depicted in Table 4.

TABLE 5. The comparison results among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM for US-based companies.

Company	Metrics	Method	Hold	Buy	Sell
Intel	Recll	PLR-CNN-LSTM	0.21	0.77	0.76
		PLR-CNN-TA	0.24	0.71	0.61
		PLR-LSTM	0.14	0.74	0.73
	Precision	PLR-CNN-LSTM	0.67	0.43	0.43
		PLR-CNN-TA	0.54	0.40	0.41
		PLR-LSTM	0.53	0.40	0.40
	F1 score	PLR-CNN-LSTM	0.32	0.55	0.55
		PLR-CNN-TA	0.34	0.51	0.49
		PLR-LSTM	0.22	0.52	0.52
Netflix	Recll	PLR-CNN-LSTM	0.17	0.75	0.85
		PLR-CNN-TA	0.09	0.75	0.76
		PLR-LSTM	0.14	0.68	0.74
	Precision	PLR-CNN-LSTM	0.63	0.44	0.43
		PLR-CNN-TA	0.60	0.38	0.38
		PLR-LSTM	0.54	0.38	0.39
	F1 score	PLR-CNN-LSTM	0.27	0.56	0.57
		PLR-CNN-TA	0.16	0.51	0.50
		PLR-LSTM	0.23	0.49	0.51
Disney	Recll	PLR-CNN-LSTM	0.18	0.77	0.76
		PLR-CNN-TA	0.21	0.64	0.72
		PLR-LSTM	0.17	0.64	0.68
	Precision	PLR-CNN-LSTM	0.63	0.41	0.44
		PLR-CNN-TA	0.52	0.39	0.42
		PLR-LSTM	0.46	0.39	0.37
	F1 score	PLR-CNN-LSTM	0.28	0.53	0.56
		PLR-CNN-TA	0.30	0.48	0.53
		PLR-LSTM	0.24	0.49	0.48

B. COMPUTATIONAL MODEL PERFORMANCE

In this section, we discuss the experimental results to demonstrate the computational model performance of the proposed model. It have been shown that CNN-TA performs very well against Buy & Hold and other models over long periods of out-of-sampletest periods [45] and LSTM is very good at processing and analysis of time series data. Therefore, in the experiments, we carry out the comparative experiments among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM

TABLE 6. The comparison results among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM for Turkey-based companies.

Company	Metrics	Method	Hold	Buy	Sell
Arcelik	Recll	PLR-CNN-LSTM	0.18	0.79	0.79
		PLR-CNN-TA	0.24	0.70	0.65
		PLR-LSTM	0.20	0.74	0.67
	Precision	PLR-CNN-LSTM	0.68	0.43	0.36
		PLR-CNN-TA	0.59	0.41	0.34
		PLR-LSTM	0.61	0.40	0.34
	F1 score	PLR-CNN-LSTM	0.29	0.56	0.50
		PLR-CNN-TA	0.34	0.52	0.45
		PLR-LSTM	0.30	0.52	0.45
Dogus	Recll	PLR-CNN-LSTM	0.12	0.84	0.73
		PLR-CNN-TA	0.13	0.79	0.70
		PLR-LSTM	0.13	0.80	0.63
	Precision	PLR-CNN-LSTM	0.66	0.31	0.37
		PLR-CNN-TA	0.68	0.30	0.34
		PLR-LSTM	0.61	0.29	0.35
	F1 score	PLR-CNN-LSTM	0.20	0.45	0.49
		PLR-CNN-TA	0.21	0.44	0.46
		PLR-LSTM	0.21	0.42	0.45
Koc Holding A.S	Recll	PLR-CNN-LSTM	0.18	0.75	0.78
		PLR-CNN-TA	0.26	0.59	0.74
		PLR-LSTM	0.24	0.63	0.68
	Precision	PLR-CNN-LSTM	0.61	0.44	0.37
		PLR-CNN-TA	0.60	0.44	0.34
		PLR-LSTM	0.61	0.42	0.33
	F1 score	PLR-CNN-LSTM	0.28	0.55	0.50
		PLR-CNN-TA	0.36	0.50	0.47
		PLR-LSTM	0.35	0.51	0.45

TABLE 7. The comparison results among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM for ETFs.

ETF	Metrics	Method	Hold	Buy	Sell
QQQ	Recll	PLR-CNN-LSTM	0.28	0.75	0.70
		PLR-CNN-TA	0.34	0.60	0.59
		PLR-LSTM	0.23	0.65	0.66
	Precision	PLR-CNN-LSTM	0.65	0.47	0.40
		PLR-CNN-TA	0.56	0.47	0.36
		PLR-LSTM	0.58	0.41	0.35
	F1 score	PLR-CNN-LSTM	0.39	0.58	0.51
		PLR-CNN-TA	0.42	0.53	0.44
		PLR-LSTM	0.33	0.50	0.46
SPY	Recll	PLR-CNN-LSTM	0.09	0.87	0.84
		PLR-CNN-TA	0.14	0.71	0.79
		PLR-LSTM	0.16	0.75	0.68
	Precision	PLR-CNN-LSTM	0.70	0.40	0.40
		PLR-CNN-TA	0.57	0.42	0.35
		PLR-LSTM	0.56	0.38	0.36
	F1 score	PLR-CNN-LSTM	0.16	0.55	0.54
		PLR-CNN-TA	0.22	0.53	0.48
		PLR-LSTM	0.25	0.51	0.47
XLP	Recll	PLR-CNN-LSTM	0.14	0.77	0.80
		PLR-CNN-TA	0.16	0.67	0.75
		PLR-LSTM	0.11	0.73	0.79
	Precision	PLR-CNN-LSTM	0.62	0.40	0.42
		PLR-CNN-TA	0.53	0.40	0.38
		PLR-LSTM	0.58	0.39	0.38
	F1 score	PLR-CNN-LSTM	0.23	0.53	0.55
		PLR-CNN-TA	0.25	0.50	0.50
		PLR-LSTM	0.19	0.51	0.51
XLU	Recll	PLR-CNN-LSTM	0.23	0.72	0.77
		PLR-CNN-TA	0.21	0.69	0.68
		PLR-LSTM	0.18	0.71	0.71
	Precision	PLR-CNN-LSTM	0.63	0.48	0.39
		PLR-CNN-TA	0.56	0.43	0.36
		PLR-LSTM	0.61	0.40	0.37
	F1 score	PLR-CNN-LSTM	0.34	0.57	0.52
		PLR-CNN-TA	0.30	0.53	0.47
		PLR-LSTM	0.27	0.51	0.49

separately for each selected stock and ETF. In PLR-LSTM, we uses two LSTM layers with 64 neurons each. The return sequence is set to true in the fisrt LSTM layer and the output is connected to a dropout layer with rate 15% to avoid the over-fitting. In the second LSTM layer, the return sequence is set to false. We also use the dropout layer with rate 15% after the final LSTM layer. Table 5, Table 6 and Table 7 list the comparison results of recall, precision and F1 score in US-based companies, Turkey-based companies and ETFs among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM on the same testing set respectively.

The results in Table 5 show that the recall values of Buy class and Sell class in PLR-CNN-LSTM is significantly higher than these in PLR-CNN-TA and PLR-LSTM for all three US-based companies. The precision of PLR-CNN-LSTM is also higher than that in PLR-CNN-TA and PLR-LSTM. However, classes Buy and Sell have worse precision values compared to class Hold. In the prediction of trading signals, the recall of Buy signals and Sell signals is more valuable than the recall of Hold signals. For stock trading systems, accurate Buy or Sell signals are very important in trading algorithms. In the experiments, the PLR-CNN-LSTM model correctly captured most of the Buy and Sell signals. However, predictive models can also generate a lot of error on Buy and Sell signals. This is mainly due to the fact that the frequency of Buy and Sell signals is much lower than that of Hold signals, when Hold signals dominate the overall distribution, it is difficult for neural networks to capture Buy signals and Sell signals. To be able to predict most Buy and Sell signals (recall), the model balances these signals by producing misleading predictions for non existent Buy and Sell signals (precision). In addition, Hold signals are not as clear as Buy and Sell signals (peaks and valleys). Neural networks are likely to confuse some Hold signals with Buy and Sell signals, especially when they are near tops or bottoms.

From Table 6 and Table 7, we can see that the performances among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM on Turkey-based companies and ETFs are similar to that in the US-based companies. It illustrates that our proposed algorithm is effective in Turkey-based companies and ETFs.

Table 8 and Table 9 list the comparison results of accuracy metrics in companies and ETFs. From the table, we can see that our proposed PLR-CNN-LSTM model gives significantly higher forecasting accuracy on all US-based companies, Turkey-based companies and ETFs than

TABLE 8. The comparison accuracy(%) results among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM for US-based and Turkey-based companies.

Methods	Stocks in US			Stocks in Turkey			Average
	Intel	Netflix	Disney	Arcelik	Dogus	Koc Holding A.S	
PLR-CNN-LSTM	46.75	46.32	45.50	43.94	37.13	43.63	43.88
PLR-CNN-TA	43.75	39.88	42.88	42.50	36.13	43.31	41.41
PLR-LSTM	41.63	40.69	39.50	41.50	35.00	42.31	40.11

TABLE 9. The comparison accuracy(%) results among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM for ETFs.

Method	QQQ	SPY	XLP	XLU	Average
PLR-CNN-LSTM	48.38	42.56	43.75	46.75	45.36
PLR-CNN-TA	45.56	40.63	41.31	42.56	42.52
PLR-LSTM	42.13	40.25	40.69	42.06	41.28

TABLE 10. Evaluation of PLR-CNN-LSTM method.

	Metrics	Hold	Buy	Sell
Stocks in US	Recll	0.19	0.76	0.79
	Precision	0.64	0.43	0.43
	F1 score	0.29	0.55	0.56
Stocks in Turkey	Recll	0.16	0.79	0.77
	Precision	0.65	0.39	0.37
	F1 score	0.26	0.52	0.50
ETFs	Recll	0.19	0.78	0.78
	Precision	0.65	0.44	0.40
	F1 score	0.28	0.56	0.53

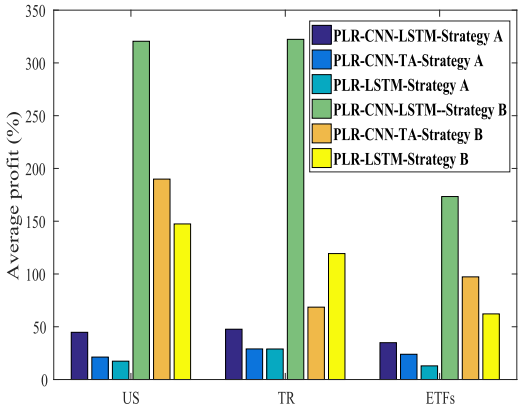

FIGURE 9. The average profit with strategy A and B.

PLR-CNN-TA and PLR-LSTM. From Table 8, we can see the average accuracy of companies for PLR-CNN-LSTM is 43.88% while it is 41.41% for PLR-CNN-TA and 40.11% for PLR-LSTM. Table 9 shows that the average accuracy of ETFs for PLR-CNN-LSTM is 45.36% while it is 42.52% for PLR-CNN-TA and 41.28% for PLR-LSTM.

Table 10 illustrates the performance evaluation of the results for US-based companies, Turkey-based companies and ETFs. In general, the recall performance of Buy signal (entry points) and Sell signal (exit points) are much better when compared with Hold signal, while it reverses in precision performance, Buy signal and Sell signal have lower precision rate compared to that of Hold. For stock trading systems, accurate entry and exit points are one of the most important factors to the overall success of the trading algorithm. From the table, we can see that most of the Buy and Sell points are captured correctly by the proposed model although some false entry and exit points generated as well. The reason is mainly due to the fact that Buy and Sell signals appear much less frequently than the Hold points, and it is difficult to catch the ''seldom'' entry and exit points without jeopardizing the general distribution of the dominant Hold values.

C. TRADING SIMULATION

In this section, we carry out some trading simulations to see the relationship between predictability and profitability with two investment strategies. Table 11, Table 12 and Table 13 list the comparison results among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM in the same testing set for the stocks in US-based companies, Turkey-based companies and ETFs, respectively.

From Table 11, we can see that in the transaction with strategy A, PLR-CNN-LSTM wins PLR-CNN-TA and PLR-LSTM on all US-based companies in the profit. The average profit of PLR-CNN-LSTM is 44.75%, which is 23.47% and 27.35% higher than that of PLR-CNN-TA and PLR-LSTM respectively. The average investment money, buying times and selling times for PLR-CNN-LSTM are 1590.15, 668.67 and 284 while they are 3591.37, 668.67 and 219.33 for PLR-CNN-TA and 2380.48, 655.33 and 239.67 for PLR-LSTM, respectively. This means that this strategy is suitable for the investors that possess a lot of funds. In the transaction with strategy B, PLR-CNN-LSTM wins PLR-CNN-TA and PLR-LSTM on all companies in the profit with the same fund while it needs buying and selling more times than PLR-CNN-TA and PLR-LSTM. The average profit, buying times and selling times for PLR-CNN-LSTM are 320.52%, 290.67 and 284 while they are 189.91%, 223.67 and 219.33 for PLR-CNN-TA and 147.43%, 244 and 239.67 for PLR-LSTM, respectively.

From Table 12, we can see the profitability with two investment strategies among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM on the Turkey-based companies are similar to that in US-based companies. In the transaction with strategies A and B, the average profits in PLR-CNN-LSTM are 47.68% and 322.34% while they are 29.00%, 68.58% for PLR-CNN-TA and 28.94%, 119.34% for PLR-LSTM. From the above results, we can see the average profits in PLR-CNN-LSTM are higher than PLR-CNN-TA with 18.68% using strategy A, 253.76% using strategy B and

TABLE 11. The financial evaluation comparison results among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM for US-based companies.

Company	Method	Strategy A				Strategy B			
		V_{money}	Profit(%)	N_{buy}	N_{sell}	V_{money}	Profit(%)	N_{buy}	N_{sell}
Intel	PLR-CNN-LSTM	397.23	48.14	673	288	100000.00	307.11	293	288
	PLR-CNN-TA	474.85	28.83	664	228	100000.00	130.04	231	228
	PLR-LSTM	688.56	11.51	693	259	100000.00	87.98	262	259
Netflix	PLR-CNN-LSTM	3182.17	44.90	625	287	100000.00	537.62	295	287
	PLR-CNN-TA	8643.30	11.58	726	220	100000.00	400.32	228	220
	PLR-LSTM	4300.24	30.77	671	239	100000.00	307.32	247	239
Disney	PLR-CNN-LSTM	1191.04	41.22	708	277	100000.00	116.83	284	277
	PLR-CNN-TA	1655.95	23.43	616	210	100000.00	39.37	212	210
	PLR-LSTM	2152.63	9.91	602	221	100000.00	46.99	223	221
Average	PLR-CNN-LSTM	1590.15	44.75	668.67	284	100000.00	320.52	290.67	284
	PLR-CNN-TA	3591.37	21.28	668.67	219.33	100000.00	189.91	223.67	219.33
	PLR-LSTM	2380.48	17.40	655.33	239.67	100000.00	147.43	244	239.67

TABLE 12. The financial evaluation comparison results among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM for Turkey-based companies.

Company	Method	Strategy A				Strategy B			
		V_{money}	Profit(%)	N_{buy}	N_{sell}	V_{money}	Profit(%)	N_{buy}	N_{sell}
Arcelik	PLR-CNN-LSTM	636.05	30.73	625	265	100000.00	321.22	268	265
	PLR-CNN-TA	496.57	27.66	585	217	100000.00	38.53	221	217
	PLR-LSTM	1069.70	19.08	628	235	100000.00	120.21	244	235
Dogus	PLR-CNN-LSTM	586.70	52.56	839	246	100000.00	289.74	251	246
	PLR-CNN-TA	642.26	16.54	795	207	100000.00	22.81	209	207
	PLR-LSTM	1738.15	37.47	847	227	100000.00	96.55	237	227
Koc Holding A.S	PLR-CNN-LSTM	732.36	59.76	597	276	100000.00	356.07	279	276
	PLR-CNN-TA	242.51	42.80	468	195	100000.00	144.40	199	195
	PLR-LSTM	350.16	30.27	527	228	100000.00	141.26	230	228
Average	PLR-CNN-LSTM	651.70	47.68	687	262.33	100000.00	322.34	266	262.33
	PLR-CNN-TA	460.45	29.00	616	206.33	100000.00	68.58	209.67	206.33
	PLR-LSTM	1052.67	28.94	667.33	230	100000.00	119.34	237	230

TABLE 13. The financial evaluation comparison results among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM for ETFs.

Code of stock	Method	Strategy A				Strategy B			
		V_{money}	Profit(%)	N_{buy}	N_{sell}	V_{money}	Profit(%)	N_{buy}	N_{sell}
QQQ	PLR-CNN-LSTM	1923.55	56.68	571	243	100000.00	273.11	243	243
	PLR-CNN-TA	2288.36	36.98	464	197	100000.00	190.50	197	197
	PLR-LSTM	2931.20	20.32	567	245	100000.00	83.96	251	245
SPY	PLR-CNN-LSTM	3786.99	30.75	744	300	100000.00	152.87	303	300
	PLR-CNN-TA	3603.46	24.72	584	210	100000.00	64.06	215	210
	PLR-LSTM	7891.55	8.83	678	238	100000.00	56.21	238	238
XLP	PLR-CNN-LSTM	555.19	22.16	690	275	100000.00	108.99	276	275
	PLR-CNN-TA	935.76	10.50	600	236	100000.00	28.14	238	236
	PLR-LSTM	662.21	11.19	668	267	100000.00	46.25	268	267
XLU	PLR-CNN-LSTM	429.55	30.13	546	264	100000.00	158.57	266	264
	PLR-CNN-TA	618.85	23.57	579	227	100000.00	106.40	228	227
	PLR-LSTM	735.82	11.45	652	241	100000.00	62.34	243	241
Average	PLR-CNN-LSTM	1673.82	34.93	637.75	270.5	100000.00	173.39	272	270.5
	PLR-CNN-TA	1861.61	23.94	556.75	217.5	100000.00	97.28	219.5	217.5
	PLR-LSTM	3055.20	12.95	641.25	247.75	100000.00	62.19	250	247.75

higher than PLR-LSTM with 18.74% using strategy A, 203% using strategy B, respectively. From the above analysis, it is clear that PLR-CNN-LSTM achieves the best performance on all companies.

In Table 13, it can be seen PLR-CNN-LSTM can earn substantially more profits than PLR-CNN-TA and PLR-LSTM in the transaction with strategy A and B. The implication of these findings is that our proposed algorithm is effective in trading with ETFs.

Figure 9 shows the average profits among PLR-CNN-LSTM, PLR-CNN-TA and PLR-LSTM utilizing strategy A and strategy B for US-based companies, Turkey-based companies and ETFs respectively.

V. CONCLUSION

In this paper, we propose a combined method by integrating PLR with CNN and LSTM neural networks to predict the trading signals of the financial market. Firstly, we use PLR method to decompose the financial data into different segments for generating the trading points (peaks and valleys). Secondly, we analyze financial time series data, convert them into 2-D images and labeled them as Buy, Sell or Hold depending on the peaks and valleys through PLR method. Thirdly, CNN-LSTM is used to model the prediction of trading points from the historical data using training data, afterwards the trained model is used to forecast the future turning points using the test data. We compare the proposed method with PLR-CNN-TA and PLR-LSTM on US-based companies, Turkey-based companies and ETFs. The experiment results clearly show that PLR-CNN-LSTM has highest forecasting performance on all companies and ETFs and can provide the highest profitability with different investment strategies.

However, there are still some problems to be studied further. Future research work will design better learning models to make the results more accurate. Future research work will explore other good investment strategies since different investment strategies have large effect on profits and an unsuitable strategy can lead to poor returns despite of the high forecasting performance.

REFERENCES

[1] L.-J. Kao, C.-C. Chiu, C.-J. Lu, and C.-H. Chang, "A hybrid approach by integrating wavelet-based feature extraction with Mars and SVR for stock index forecasting," *Decis. Support Syst.*, vol. 54, no. 3, pp. 1228–1244, Feb. 2013.

[2] W. Shen, X. Guo, C. Wu, and D. Wu, "Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm," *Knowl.-Based Syst.*, vol. 24, pp. 378–385, 2011.

[3] Y.-J. Chen, Y.-M. Chen, and C. L. Lu, "Enhancement of stock market forecasting using an improved fundamental analysis-based approach," *Soft Comput.*, vol. 21, no. 13, pp. 3735–3757, Jul. 2017.

[4] N. Mohamed, "Technical analysis or fundamental analysis impacts on the Moroccan stock market: Analyse technique ou analyse fondamentale impacts sur le marché boursier marocain," *Afr. Sci. J.*, vol. 3, p. 234, Jun. 2021.

[5] V. K. Bogullu, D. Enke, and C. H. Dagli, "Using neural networks and technical indicators for generating stock trading signals," in *Intelligent Engineering Systems Through Artificial Neural Networks* (American Society of Mechanical Engineers), vol. 12. 2022, pp. 721–726.

[6] Y. Shynkevich, T. M. McGinnity, S. A. Coleman, A. Belatreche, and Y. Li, "Forecasting price movements using technical indicators: Investigating the impact of varying input window length," *Neurocomputing*, vol. 264, pp. 71–88, Nov. 2017.

[7] R. D. Edwards, J. Magee, and W. C. Bassetti, *Technical Analysis of Stock Trends*. Boca Raton, FL, USA: CRC Press, 2018.

[8] R. P. Rustagi, *Investment Analysis & Portfolio Management*. Chennai, India: Sultan Chand & Sons, 2021.

[9] S. Walczak and V. Velanovich, "Improving prognosis and reducing decision regret for pancreatic cancer treatment using artificial neural networks," *Decis. Support Syst.*, vol. 106, pp. 110–118, Feb. 2018.

[10] A. Kurani, P. Doshi, A. Vakharia, and M. Shah, "A comprehensive comparative study of artificial neural network (ANN) and support vector machines (SVM) on stock forecasting," *Ann. Data Sci.*, vol. 10, no. 1, pp. 183–208, 2021.

[11] M. Ozcalici and M. Bumin, "Optimizing filter rule parameters with genetic algorithm and stock selection with artificial neural networks for an improved trading: The case of Borsa Istanbul," *Expert Syst. Appl.*, vol. 208, Dec. 2022, Art. no. 118120.

[12] S. Piri, D. Delen, and T. Liu, "A synthetic informative minority over-sampling (SIMO) algorithm leveraging support vector machine to enhance learning from imbalanced datasets," *Decis. Support Syst.*, vol. 106, pp. 15–29, Feb. 2018.

[13] A. Cheng, X. Jiang, Y. Li, C. Zhang, and H. Zhu, "Multiple sources and multiple measures based traffic flow prediction using the chaos theory and support vector regression method," *Phys. A, Statist. Mech. Appl.*, vol. 466, pp. 422–434, Jan. 2017.

[14] Y. Chen and Y. Hao, "A feature weighted support vector machine and K-nearest neighbor algorithm for stock market indices prediction," *Expert Syst. Appl.*, vol. 80, pp. 340–355, Sep. 2017.

[15] Y. Chen and Y. Hao, "A novel framework for stock trading signals forecasting," *Soft Comput.*, vol. 24, no. 16, pp. 12111–12130, Aug. 2020.

[16] P.-Y. Hao, C.-F. Kung, C.-Y. Chang, and J.-B. Ou, "Predicting stock price trends based on financial news articles and using a novel twin support vector machine with fuzzy hyperplane," *Appl. Soft Comput.*, vol. 98, Jan. 2021, Art. no. 106806.

[17] W. Sun and J. Zhang, "A novel carbon price prediction model based on optimized least square support vector machine combining characteristic-scale decomposition and phase space reconstruction," *Energy*, vol. 253, Aug. 2022, Art. no. 124167.

[18] B. B. Nair, V. P. Mohandas, and N. R. Sakthivel, "A decision tree-rough set hybrid system for stock market trend prediction," *Int. J. Comput. Appl.*, vol. 6, no. 9, pp. 1–6, Sep. 2010.

[19] B. Sun, W. Ma, and Y. Qian, "Multigranulation fuzzy rough set over two universes and its application to decision making," *Knowl.-Base Syst.*, vol. 123, pp. 61–74, May 2017.

[20] R. K. Nayak, R. Tripathy, D. Mishra, V. K. Burugari, P. Selvaraj, A. Sethy, and B. Jena, "Indian stock market prediction based on rough set and support vector machine approach," in *Intelligent and Cloud Computing*. Cham, Switzerland: Springer, 2021, pp. 345–355.

[21] M. H. Abolbashari, E. Chang, O. K. Hussain, and M. Saberi, "Smart buyer: A Bayesian network modelling approach for measuring and improving procurement performance in organisations," *Knowl.-Based Syst.*, vol. 142, pp. 127–148, Feb. 2018.

[22] K. Topuz, F. D. Zengul, A. Dag, A. Almehmi, and M. B. Yildirim, "Predicting graft survival among kidney transplant recipients: A Bayesian decision support model," *Decis. Support Syst.*, vol. 106, pp. 97–109, Feb. 2018.

[23] B. Baba and G. Sevil, "Bayesian analysis of time-varying interactions between stock returns and foreign equity flows," *Financial Innov.*, vol. 7, no. 1, pp. 1–25, Dec. 2021.

[24] J. Zhang and Y.-M. Zhuang, "Cross-market infection research on stock herding behavior based on DGC-MSV models and Bayesian network," *Complexity*, vol. 2021, pp. 1–8, Jan. 2021.

[25] J. Zhao, L. Jiao, S. Xia, V. B. Fernandes, I. Yevseyeva, Y. Zhou, and M. T. M. Emmerich, "Multiobjective sparse ensemble learning by means of evolutionary algorithms," *Decis. Support Syst.*, vol. 111, pp. 86–100, Jul. 2018.

[26] Z. Meng and J.-S. Pan, "Monkey king evolution: A new memetic evolutionary algorithm and its application in vehicle fuel consumption optimization," *Knowl.-Based Syst.*, vol. 97, pp. 144–157, Apr. 2016.

[27] W. Sheng, P. Shan, S. Chen, Y. Liu, and F. E. Alsaadi, "A niching evolutionary algorithm with adaptive negative correlation learning for neural network ensemble," *Neurocomputing*, vol. 247, pp. 173–182, Jul. 2017.

[28] R. Kumar, P. Kumar, and Y. Kumar, "Two-phase hybridisation using deep learning and evolutionary algorithms for stock market forecasting," *Int. J. Grid Utility Comput.*, vol. 12, nos. 5–6, pp. 573–589, 2021.

[29] A. Canziani, A. Paszke, and E. Culurciello, "An analysis of deep neural network models for practical applications," 2016, *arXiv:1605.07678*.

[30] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[31] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," 2014, *arXiv:1412.5068*.

[32] S. Kim, S. Hong, M. Joh, and S.-K. Song, "DeepRain: ConvLSTM network for precipitation prediction using multichannel radar data," 2017, *arXiv:1711.02316*.

[33] S. Dargan, M. Kumar, M. R. Ayyagari, and G. Kumar, "A survey of deep learning and its applications: A new paradigm to machine learning," *Arch. Comput. Methods Eng.*, vol. 27, no. 4, pp. 1071–1092, Sep. 2019.

[34] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Comput. Sci. Rev.*, vol. 40, May 2021, Art. no. 100379.

[35] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014.

[36] X. Liu, L. Song, S. Liu, and Y. Zhang, "A review of deep-learning-based medical image segmentation methods," *Sustainability*, vol. 13, no. 3, p. 1224, Jan. 2021.

[37] F. Isensee, P. F. Jaeger, S. A. A. Kohl, J. Petersen, and K. H. Maier-Hein, "nnU-Net: A self-configuring method for deep learning-based biomedical image segmentation," *Nature Methods*, vol. 18, no. 2, pp. 203–211, Dec. 2020.

[38] G. Hinton, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[39] W. Lee, J. J. Seong, B. Ozlu, B. S. Shim, A. Marakhimov, and S. Lee, "Biosignal sensors and deep learning-based speech recognition: A review," *Sensors*, vol. 21, no. 4, p. 1399, Feb. 2021.

[40] L. A. Kumar, D. K. Renuka, S. L. Rose, M. C. Shunmuga Priya, and I. M. Wartana, "Deep learning based assistive technology on audio visual speech recognition for hearing impaired," *Int. J. Cognit. Comput. Eng.*, vol. 3, pp. 24–30, Jun. 2022.

[41] C. D. Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts," in *Proc. COLING 25th Int. Conf. Comput. Linguistics, Tech. Papers*, 2014, pp. 69–78.

[42] I. Lauriola, A. Lavelli, and F. Aiolli, "An introduction to deep learning in natural language processing: Models, techniques, and tools," *Neurocomputing*, vol. 470, pp. 443–456, Jan. 2022.

[43] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[44] H. Ren, L. Sun, J. Guo, C. Han, and Y. Cao, "A high compatibility finger vein image quality assessment system based on deep learning," *Expert Syst. Appl.*, vol. 196, Jun. 2022, Art. no. 116603.

[45] O. B. Sezer and A. M. Ozbayoglu, "Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach," *Appl. Soft Comput.*, vol. 70, pp. 525–538, Apr. 2018.

[46] Z. D. Aksehir and E. Kilic, "How to handle data imbalance and feature selection problems in CNN-based stock price forecasting," *IEEE Access*, vol. 10, pp. 31297–31305, 2022.

[47] Y. Venkateswarlu, K. Baskar, A. Wongchai, V. G. Shankar, C. P. M. Carranza, J. L. A. Gonzáles, and A. R. M. Dharan, "An efficient outlier detection with deep learning-based financial crisis prediction model in big data environment," *Comput. Intell. Neurosci.*, vol. 2022, pp. 1–10, Aug. 2022.

[48] Y. Cao, Y. Shao, and H. Zhang, "Study on early warning of E-commerce enterprise financial risk based on deep learning algorithm," *Electron. Commerce Res.*, vol. 22, no. 1, pp. 21–36, Mar. 2022.

[49] E. Hoseinzade, S. Haratizadeh, and A. Khoeini, "U-CNNpred: A universal CNN-based predictor for stock markets," 2019, *arXiv:1911.12540*.

[50] A. Kelotra and P. Pandey, "Stock market prediction using optimized deep-ConvLSTM model," *Big Data*, vol. 8, no. 1, pp. 5–24, Feb. 2020.

[51] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*, vol. 3361, no. 10. Cambridge, MA, USA: MIT Press, 1995, p. 1995.

[52] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[53] C. J. Neely, D. E. Rapach, J. Tu, and G. Zhou, "Forecasting the equity risk premium: The role of technical indicators," *Manage. Sci.*, vol. 60, no. 7, pp. 1772–1791, 2014.

[54] C.-C. Lin, C.-L. Lin, and J. Z. Shyu, "Hybrid multi-model forecasting system: A case study on display market," *Knowl.-Based Syst.*, vol. 71, pp. 279–289, Nov. 2014.

[55] S. Panigrahi and H. S. Behera, "A hybrid ETS–ANN model for time series forecasting," *Eng. Appl. Artif. Intell.*, vol. 66, pp. 49–59, Nov. 2017.

[56] A. Nikfarjam, E. Emadzadeh, and S. Muthaiyah, "Text mining approaches for stock market prediction," in *Proc. 2nd Int. Conf. Comput. Autom. Eng. (ICCAE)*, Feb. 2010, pp. 256–260.

[57] P.-C. Chang, T. W. Liao, J.-J. Lin, and C.-Y. Fan, "A dynamic threshold decision system for stock trading signal detection," *Appl. Soft Comput.*, vol. 11, no. 5, pp. 3998–4010, Jul. 2011.

[58] L. Luo, S. You, Y. Xu, and H. Peng, "Improving the integration of piece wise linear representation and weighted support vector machine for stock trading signal prediction," *Appl. Soft Comput.*, vol. 56, pp. 199–216, Jul. 2017.

[59] L. Luo and X. Chen, "Integrating piecewise linear representation and weighted support vector machine for stock trading signal prediction," *Appl. Soft Comput.*, vol. 13, no. 2, pp. 806–816, Feb. 2013.

[60] Y. LeCun, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.

[61] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 2, pp. 84–90, Jun. 2012.

[62] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1725–1732.

[63] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in *Proc. 29th AAAI Conf. Artif. Intell.*, vol. 333, 2015, pp. 2267–2273.

[64] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[65] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.*, Cham, Switzerland: Springer, 2016, pp. 630–645.

[66] X. Liu, K. Yan, L. Burak Kara, and Z. Nie, "CCFD-Net: A novel deep learning model for credit card fraud detection," in *Proc. IEEE 22nd Int. Conf. Inf. Reuse Integr. Data Sci. (IRI)*, Aug. 2021, pp. 9–16.